Shape Your Dreams, Build Your Future!

BIGDATA

PLACEMENT ORIENTED!

- Online Live Digital Board
- 1:1 Mentorship
- 600+ Coding Exercises
- Placement Assistance Program
- Interview Preparation program

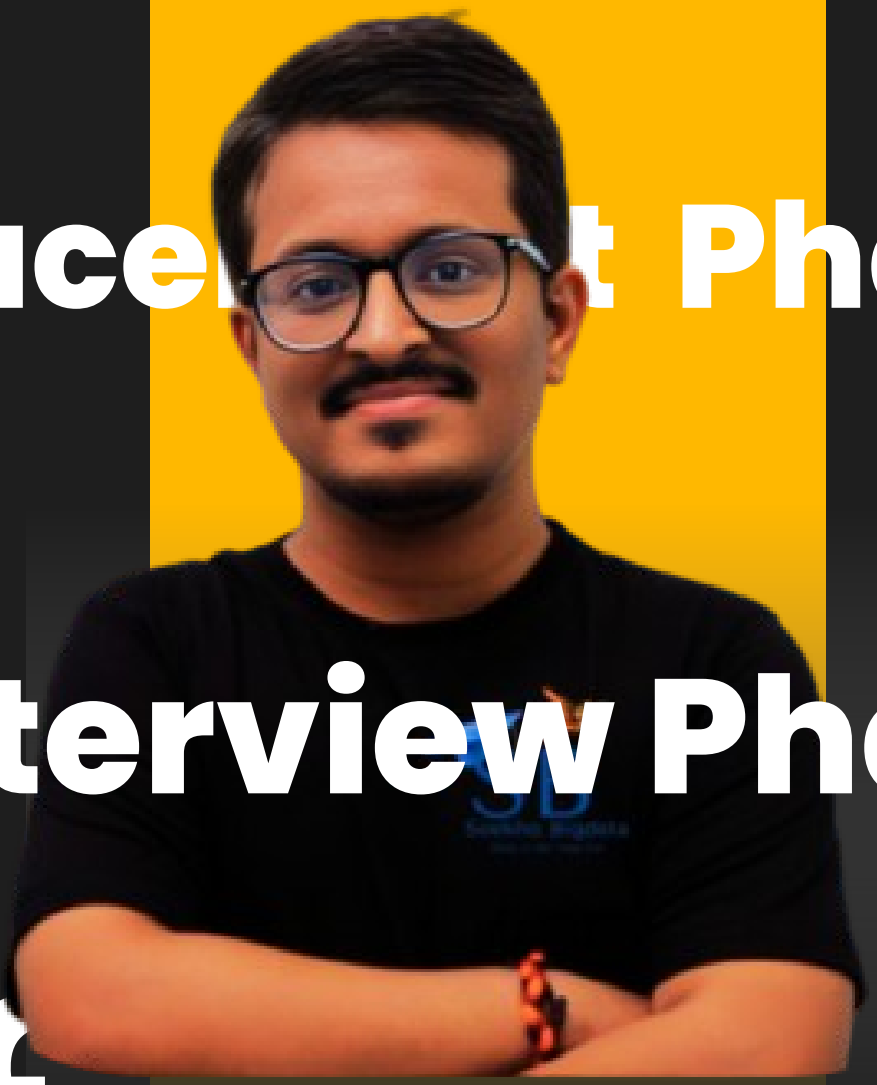
Data is New Oil

WE just dont teach we believe Teaching with Results

For more info:

+91-9989454737 www.seekhobigdata.com Hyderabad,Telangana

Learning Phase



Placement Phase

Interview Phase

Three Different Phases will help u to upskill urself With in 4 month and grab 100-500% Hike

www.seekhobigdata.com

TABLE OF CONTENTS

01.

INTRODUCTION

—

Hadoop && HDFS

02.

INGESTION TOOL

—

SQOOP && MAPREDUCE

03.

SCRIPTING

—

Shell Scripting

04.

DATABASE

—

MYSQL

05.

DATAWAREHOUSE

—

HIVE

06.

PROGRAMMING

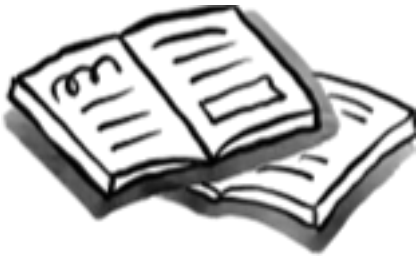
—

SCALA

TABLE OF CONTENTS

TABLE OF CONTENTS

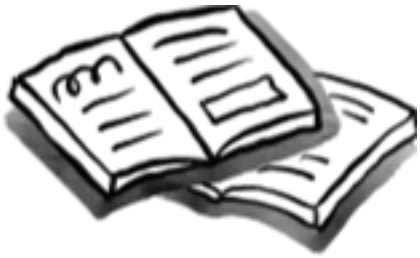
1.Learning Phase



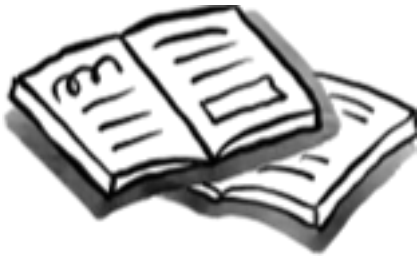
Module-1

BIGDATA &&HADOOP AND HDFS

- ☐ WHAT IS BIGDATA?
- ☐ WHAT ARE 5 V'S OF BIGDATA?
- ☐ WHAT ARE MONOLITHIC SYSTEMS?
- ☐ WHAT ARE DISTRIBUTED SYSTEMS?
- ☐ EVOLUTION OF HADOOP HOW HADOOP HAS BEEN ACCEPTED AS
- ☐ SOLUTION FOR BIGDATA ?
- ☐ WHAT IS HDFS?
- ☐ WHAT IS A BLOCK AND BLOCK SIZE?
- ☐ FIXED SIZE VS. VARIABLE SIZE BLOCK
- ☐ HOW TO TWEAK BLOCK SIZE USING FIXED AND VARIABLE BLOCK SIZE
- ☐ WHAT IS MEANT BY NODE? WHAT IS A CLUSTER?
- ☐ WHAT IS RESOURCE?
- ☐ NAMENODE DATANODE HDFS ARCHITECTURE DATANODE FAILURE MANAGEMENT



- [ ] REASONS FOR DATANODE FAILURE
- [ ] HEART-BEAT MECHANISM
- [ ] REPLICATION MECHANISM
- [ ] PROPERTIES TO CHECK: HEARTBEAT MECHANISM AND REPLICATION
- [ ] NAMENODE FAILURE MANAGEMENT
- [ ] REASONS FOR NAMENODE FAILURE
- [ ] WHAT IS ZOOKEEPER?
- [ ] SECONDARY NAMENODE
- [ ] FS IMAGE AND EDIT LOGS
- [ ] CHECKPOINTING
- [ ] FAULT TOLERANCE
- [ ] WHAT IS A SPLIT-BRAIN SCENARIO?
- [ ] WHAT IS FENCING?
- [ ] COMMAND TO KILL SNN
- [ ] PROPERTY FOR CHECKPOINTING
- [ ] CHECKPOINTING FREQUENCY

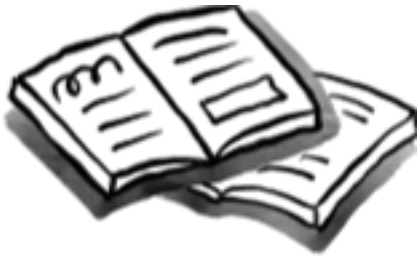


SECURITY FOR HDFS SECURİTY FOR DATA STORED İN HDFS

- [ ] WHAT IS THE KERBEROS AUTHENTICATION MECHANISM IN HDFS?
- [ ] WHAT IS KEYTAB IN HDFS?
- [ ] WHAT IS THE KNIT COMMAND?
- [ ] WHAT IS PBRUN?

CORRUPTED RECORDS IN HDFS AND HOW THEY ARE HANDLED

- [ ] HOW DATA CAN BE CORRUPTED
- [ ] HOW TO IDENTIFY CORRUPTED RECORDS IN HDFS
- [ ] WHAT IS CHECKSUM
- [ ] WHAT IS THE FSCK COMMAND
- [ ] WHAT IS THE RESOLUTION TECHNIQUE FOR DATA CORRUPTION IN HDFS

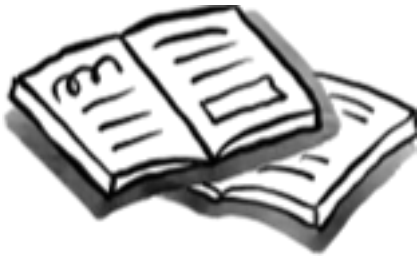


LINUX COMMANDS

- [ ] LS COMMAND AND ITS FLAVOURS
- [ ] CAT COMMAND AND ITS FLAVOURS
- [ ] MKDIR COMMAND AND ITS FLAVORS
- [ ] RM COMMAND AND ITS FLAVORS
- [ ] MV COMMAND AND ITS FLAVORS
- [ ] CP COMMAND AND ITS FLAVORS
- [ ] TAIL COMMAND
- [ ] GREP COMMAND AND ITS FLAVORS
- [ ] FIND COMMAND AND ITS FLAVORS

Vİ EDİTORCOMMANDS

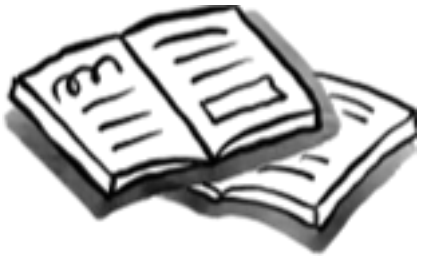
- [ ] VIEW A FILENAME: VI
- [ ] INSERT MODE: I
- [ ] SAVE WHEN UPDATED: :W
- [ ] QUIT: :Q!
- [ ] UPDATE, SAVE, AND QUIT: :WQ



FİLE PERMİSSİON COMMANDS

- [ ] WHAT ARE READ PERMISSION
- [ ] WHAT IS WRITE PERMISSION
- [ ] WHAT IS EXECUTE PERMISSION
- [ ] OWNER && GROUP AND OTHERS
- [ ] HDFS COMMANDS && PRACTICALS && SCENARIO BASED QUESTIONS

Module-2

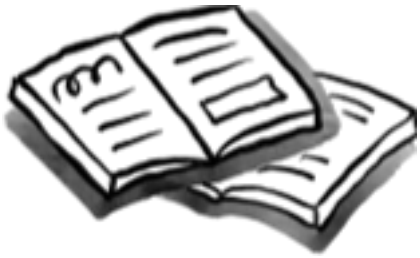


MAP REDUCE

- [ ] WHAT IS MAPREDUCE?
- [ ] WHAT ARE MAPPERS?
- [ ] WHAT IS REDUCER?
- [ ] HOW MAPREDUCE WORKS
- [ ] WHAT IS RECORDREADER?
- [ ] WHAT IS COMBINER?
- [ ] WHAT IS SHUFFLING?
- [ ] WHAT IS SORTING? LIMITATIONS OF MAPREDUCE

SQOOP

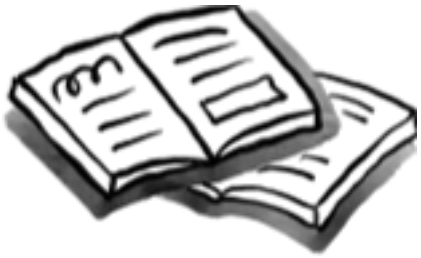
- [ ] WHAT IS SQOOP?
- [ ] WHAT IS JDBC DRIVER IN SQOOP?
- [ ] LIST-DATABASES ARGUMENT
- [ ] LIST-TABLES ARGUMENT
- [ ] IMPORT ARGUMENT
- [ ] IMPORTING SELECTED COLUMNS
- [ ] INCREASING PARALLELISM
- [ ] --NUM-MAPPERS
- [ ] SPLIT-BY ARGUMENT



SQOOP

- [ ] NULL-NON-STRING
- [ ] NULL-STRING EXPORT ARGUMENT
- [ ] STAGING IN SQOOP
- [ ] QUERY ARGUMENT
- [ ] HOW TO DECIDE MAPPERS
- [ ] HOW TO INCLUDE TABLES
- [ ] HOW TO EXCLUDE TABLES
- [ ] OUTLIERS
- [ ] INCREMENTAL LOADING IN SQOOP
- [ ] APPEND MODE
- [ ] LAST MODIFIED MODE
- [ ] DIRECT ARGUMENT
- [ ] SCENARIO-BASED QUESTIONS

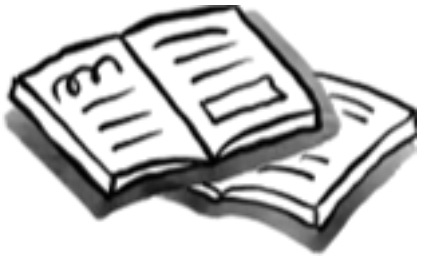
Module-3



SHELL SCRİPTİNG

- [ ] ECHO
- [ ] READONLY
- [ ] ARRAYS
- [ ] STRING
- [ ] ARITHMETIC OPERATIONS
- [ ] LOOPS
- [ ] FOR LOOP
- [ ] WHILE LOOP
- [ ] FUNCTIONS
- [ ] BUILDING SCRIPTS
- [ ] BUILDING SQOOP SCRIPTS
- [ ] HIVE SCRIPTS
- [ ] CRON TAB
- [ ] ORCHESTRATING SCRIPTS
- [ ] AWK
- [ ] SED
- [ ] PRACTICAL LAB: BUILDING 50+ SCRIPTS

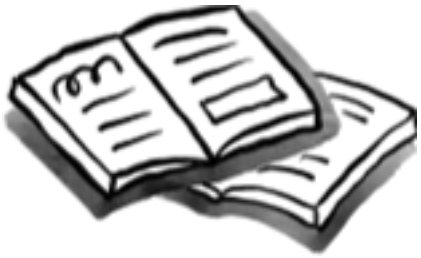
Module-4



MYSQL

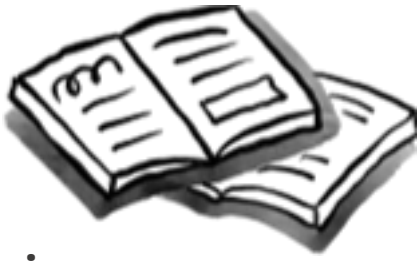
- [ ] CREATE
- [ ] ALTER
- [ ] DELETE
- [ ] DROP
- [ ] TRUNCATE
- [ ] GROUP BY
- [ ] HAVING
- [ ] CASE AND WHEN
- [ ] WINDOW FUNCTIONS
- [ ] RANK/DENSE RANK/ROW NUMBER
- [ ] NULL HANDLING
- [ ] DATE FUNCTIONS
- [ ] JOINS (2, 3, 4 TABLE JOINS)
- [ ] SELF JOIN
- [ ] MIN
- [ ] MAX
- [ ] AVG
- [ ] COUNT
- [ ] SUM
- [ ] STRING FUNCTIONS AND NUMBER MANIPULATION FUNCTIONS
- [ ] CTE EXPRESSIONS AND SQL OPTIMIZATIONS
- [ ] LAB PRACTICAL: 100 ASSIGNMENT QUESTIONS

Module - 5



HİVE

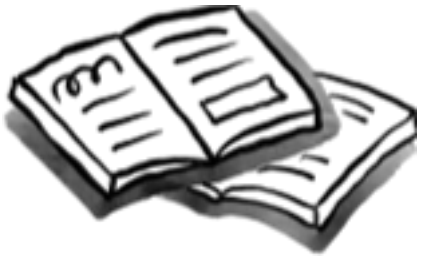
- [ ] WHAT IS A DATA WAREHOUSE?
- [ ] WHAT IS THE DIFFERENCE BETWEEN OLTP AND OLAP?
- [ ] WHAT IS HIVE?
- [ ] HOW IS SCHEMA AND DATA STORED IN HIVE?
- [ ] INTERNAL/MANAGED TABLES
- [ ] EXTERNAL TABLES
- [ ] PRIMARY DATA TYPES IN HIVE
- [ ] COMPLEX DATA TYPES IN HIVE
- [ ] ARRAY AND ITS FUNCTIONS
- [ ] MAP AND ITS FUNCTIONS
- [ ] STRUCT AND ITS FUNCTIONS
- [ ] DIFFERENT WAYS OF INSERTING DATA
- [ ] VECTORIZATION
- [ ] MSCK REPAIR
- [ ] VIEWS



OPTİMİZATİON TECHNİQUES İN HİVE

- [ ] PARTITIONING IN HIVE
- [ ] STATIC VS. DYNAMIC PARTITIONING
- [ ] BUCKETING
- [ ] HOW TO CALCULATE NUMBER OF BUCKETS
- [ ] MAP-SIDE JOIN
- [ ] BUCKET MAP JOIN
- [ ] SORT-MERGE BUCKET JOIN
- [ ] WINDOW FUNCTIONS
- [ ] RANK()
- [ ] DENSE RANK() AND ROW NUMBER()
- [ ] LEAD AND LAG
- [ ] PROBLEMS ON LEAD AND LAG AND SCD TYPES
- [ ] COMPRESSION TECHNIQUES
- [ ] MISCELLANEOUS CONCEPTS
- [ ] SCENARIO-BASED QUESTIONS
- [ ] PRACTICALS ON EVERY TOPIC

Module - 6



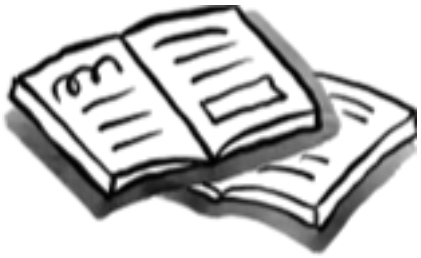
SCALA

- [ ] DATA TYPES IN SCALA
- [ ] CONDITIONAL STATEMENTS IN SCALA
- [ ] LOOPS IN SCALA
- [ ] DATA STRUCTURES IN SCALA
- [ ] ARRAYS
- [ ] MAP
- [ ] SET
- [ ] RANGE
- [ ] LIST
- [ ] TUPLE

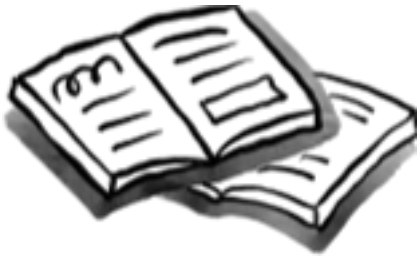
FUNCTİONAL PROGRAMMİNG

- [ ] FIRST-CLASS FUNCTIONS
- [ ] HIGHER-ORDER FUNCTIONS
- [ ] ANONYMOUS FUNCTIONS
- [ ] CLOSURES
- [ ] NULL, NIL, NOTHING, NONE, UNIT
- [ ] 50+ PRACTICAL PROBLEMS

SCALA



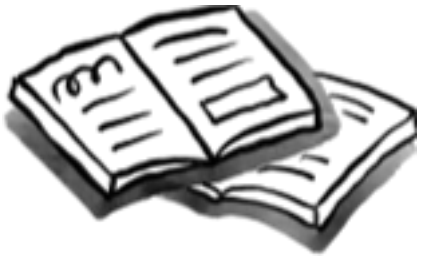
- [ ] 1.WHAT IS APP IN SCALA? WHY DO WE USE IT?
- [ ] 2.WHAT IS A SINGLETON OBJECT IN SCALA?
- [ ] 3.WHAT IS A COMPANION OBJECT?
- [ ] 4.WHAT IS AN AUXILIARY CONSTRUCTOR?
- [ ] 5.WHAT ARE IMPLICITS IN SCALA? EXPLAIN IMPLICIT VARIABLE, IMPLICIT FUNCTIONS, AND IMPLICIT CONVERSIONS?
- [ ] 6.WHAT ARE INLINE FUNCTIONS, INLINE VARIABLES IN SCALA?
- [ ] 7.WHAT ARE STREAMS IN SCALA? HOW THEY IMPROVE THE CODE EFFICIENCY?
- [ ] 8.WHAT IS A BITSET IN SCALA?
- [ ] 9.WHAT ARE MONADS IN SCALA?
- [ ] 10.HOW DO YOU INCREASE PARALLELISM IN SCALA PROGRAMMING?
- [ ] 11.WHAT IS FUTURE IN SCALA? EXPLAIN USE CASES WHERE DO WE NEED FUTURE?
- [ ] 12.WHAT IS AN OPTION?
- [ ] 13.WHAT ARE CLOSURES IN SCALA?
- [ ] 14.WHAT IS A CURRYING FUNCTION IN SCALA?
- [ ] 15.WHY THERE IS NO “STATIC” IN SCALA?
- [ ] 16.WHAT IS OFDIM() IN SCALA?
- [ ] 17.WHAT IS TAIL RECURSION IN SCALA? EXPLAIN NORMAL RECURSION VS TAIL RECURSION?
- [ ] 18.WHAT ARE SCALA TRAITS?
- [ ] 19.WHAT ARE TOUPLES IN SCALA? WHAT IS THE MAX NUMBER OF ELEMENTS A TOUPLE CAN STORE?
- [ ] 20.WHAT IS AWAIT() IN SCALA? WHY IS IT USED?



SCALA-OOPS

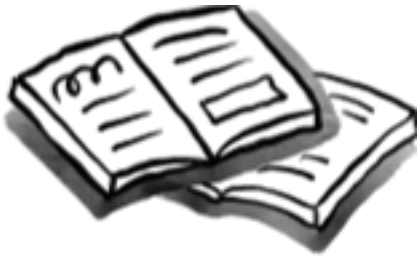
- [ ] CLASS AND OBJECT
- [ ] CONSTRUCTOR
- [ ] POLYMORPHISM
- [ ] ENCAPSULATION
- [ ] ABSTRACT CLASS
- [ ] ACCESS MODIFIERS
- [ ] DESIGN PATTERNS IN SCALA
- [ ] TRAITS
- [ ] DIAMOND PROBLEM
- [ ] CASE CLASSES
- [ ] METHOD OVERLOADING & OVERRIDING
- [ ] SINGLETON OBJECT
- [ ] COMPANION CLASSES
- [ ] 40-50 PRACTICAL PROBLEMS

Module - 7



PYTHON

- [ ] VARIABLES AND DATA TYPES
- [ ] CONTROL STRUCTURES AND LOOPS
- [ ] OPERATORS
- [ ] EXCEPTION HANDLING
- [ ] PYTHON BUILT-IN FUNCTIONS
- [ ] LISTS
- [ ] TUPLES
- [ ] SETS
- [ ] DICTIONARIES
- [ ] CLASSES
- [ ] OBJECTS
- [ ] INHERITANCE
- [ ] ENCAPSULATION
- [ ] POLYMORPHISM
- [ ] OPENING FILES
- [ ] PRIME NUMBER
- [ ] REVERSE A NUMBER
- [ ] PALINDROME
- [ ] SQUARE ROOT OF A NUMBER

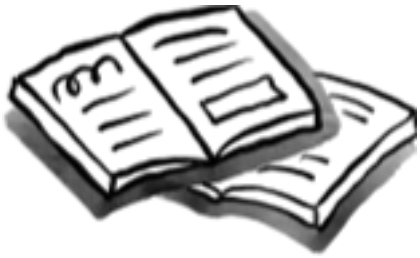


- [ ] DIVISIBILITY RULES
- [ ] MISSING NUMBER
- [ ] READING FILES
- [ ] WRITING FILES
- [ ] CLOSING FILES
- [ ] EXCEPTION HANDLING
- [ ] NUMPY
- [ ] PANDAS

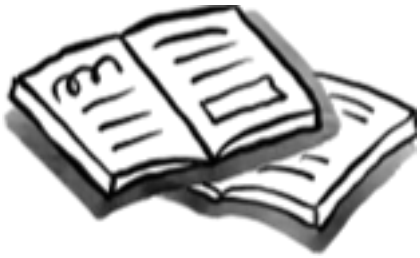
Module - 8

PY-SPARK && SCALA SPARK && SPARK-SQL

- [ ] WHAT IS APACHE SPARK?
- [ ] WHAT IS RDD?
- [ ] MAPREDUCE VS APACHE SPARK
- [ ] HOW DATA IS STORED IN SPARK
- [ ] WHAT IS IMMUTABILITY OF RDD?
- [ ] WHAT IS RESILIENT DISTRIBUTED DATASET (RDD)?
- [ ] SPARK SESSION
- [ ] SPARK CONTEXT
- [ ] PARALLELIZE()
- [ ] READ CSV, TEXTFILE

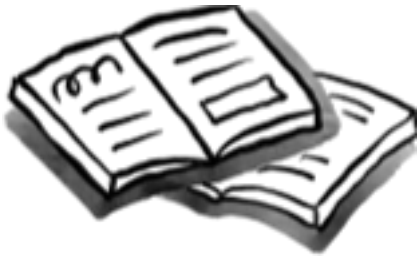


- [ ] LAZY EVALUATION
- [ ] WHAT IS DAG?
- [ ] WHAT IS LINEAGE GRAPH?
- [ ] TRANSFORMATIONS
- [ ] FAULT TOLERANCE
- [ ] LINEAGE
- [ ] MAP()
- [ ] FILTER()
- [ ] REDUCE VS REDUCEBYKEY
- [ ] GROUPBYKEY VS REDUCEBYKEY
- [ ] REPARTITION & COALESCE
- [ ] SORTBYKEY()
- [ ] FLATMAP()
- [ ] SPLIT MEAN()
- [ ] JOINS IN RDD
- [ ] CONTAINS()
- [ ] PARALLELIZE
- [ ] SPARK ARCHITECTURE
- [ ] BROADCAST VARIABLES
- [ ] ACCUMULATORS
- [ ] PROBLEMS ON RDD 60-70 PRACTICAL PROBLEMS



DATAFRAMES

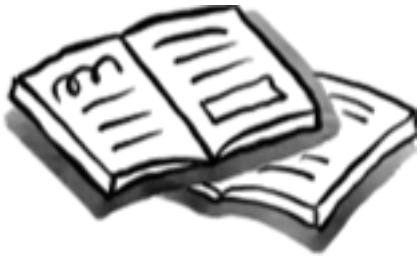
- [ ] DATAFRAMES
- [ ] DATASETS
- [ ] DATAFRAME VS DATASET
- [ ] READER API
- [ ] READ MODES
- [ ] WRITER API
- [ ] WRITE MODES
- [ ] INFER SCHEMA
- [ ] EXPLICIT SCHEMA
- [ ] DATA TYPES IN SPARK
- [ ] CONDITIONAL STATEMENTS IN SPARK
- [ ] WHEN AND OTHERWISE
- [ ] FILTER
- [ ] STRING MANIPULATION FUNCTIONS
- [ ] AGGREGATIONS
    - COUNT()
    - MIN()
    - AVG()
    - SUM
- [ ] GROUPBY AGGREGATIONS
- [ ] WINDOW AGGREGATIONS



- [ ] JOINS
  - DIFFERENT KINDS OF JOINS
  - DIFFERENT JOIN STRATAGIES
- [ ] LOG4J MECHANISM
- [ ] DIFFERENT WAYS OF DEBUGGING
- [ ] LEAD AND LAG RELATED PROBLEMS
- [ ] SPARK-SQL DATE MANIPULATION FUNCTIONS
- [ ] PRACTICALS ON EVERY CONCEPT
- [ ] BENCHMARKING TO UNDERSTAND PERFORMANCE
- [ ] STRING MANIPULATION FUNCTIONS
- [ ] NUMBER MANIPULATION FUNCTIONS
- [ ] DATA VALIDATION
- [ ] 400+ WIDE VARIETY OF PROBLEMS

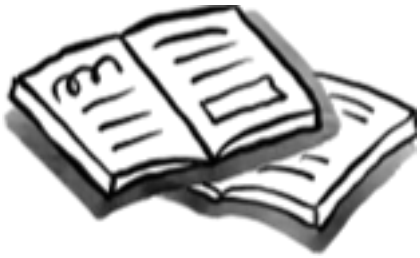
OPTİMİZATİONS

- [ ] SERIALIZATION API SELECTION
- [ ] USING BROADCAST VARIABLES
- [ ] CACHE AND PERSIST
- [ ] BYKEY OPERATION
- [ ] PREDICATE PUSHDOWN
- [ ] BROADCAST JOIN
- [ ] PARTITION AND BUCKET
- [ ] GARBAGE COLLECTION TUNING
- [ ] LEVEL OF PARALLELISM



SPARK-İSSUES

- [ ] OUT OF MEMORY EXCEPTIONS
- [ ] MISSING DATA
- [ ] DATA SKEWNESS
- [ ] SPARK JOB REPEATEDLY FAILS
- [ ] INFERSCHEMA ISSUE
- [ ] SLOW PERFORMANCE ISSUES
- [ ] MEMORY CONTENTION
- [ ] DISK CONTENTION
- [ ] BROADCASTING LARGE DATA
- [ ] SERIALIZATION ISSUE
- [ ] VERSION INCOMPATIBILITY ISSUE
- [ ] CLUSTER INSTABILITY ISSUES
- [ ] SMALL FILE ISSUE
- [ ] RESULT EXCEEDS DRIVER MEMORY
- [ ] TOO SMALL AND LARGE PARTITIONS



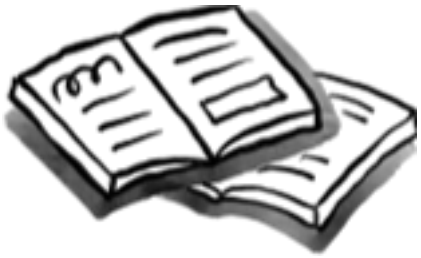
SPARK- DEPLOYEMENT

- [ ] BUILD TOOLS
- [ ] SBT BUILD TOOL
- [ ] GRADLE BUILD TOOL
- [ ] MAVEN BUILD TOOL
- [ ] JFROG
- [ ] JIRA TOOL
- [ ] BITBUCKET
- [ ] GITHUB
- [ ] GIT COMMANDS
- [ ] HOW TO BUILD A JAR
- [ ] SPARK-SUBMIT
- [ ] PARAMETERS OF SPARK-SUBMI

DATA- QUALİTYCHECKS AND DATA VALIDATIONS

- [ ] CHECK FOR DUPLICATES
- [ ] CHECK FOR UNIQUE VALUES IN COLUMNS
- [ ] CHECK FOR MISSING VALUES
- [ ] FIND OUTLIERS
- [ ] SCHEMA VALIDATION
- [ ] CORRELATIONS
- [ ] CROSS-FIELD VALIDATION
- [ ] DEPENDENCY CHECK
- [ ] TEXT PATTERN ANALYSIS
- [ ] CATEGORICAL VALUE DISTRIBUTIONS

Module - 9

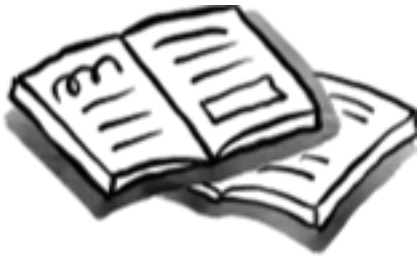


AWS S3 BASİCS

- [ ] WHAT IS AWS ?
- [ ] AWS GUI WALKTHROUGH ?
- [ ] WHAT IS REGION?
- [ ] WHAT IS EDGE LOCATION ?
- [ ] WHAT IS AVAILABILITY ZONE AND LOCAL ZONE?
- [ ] WHAT IS MULTIREGION CONCEPT ?
- [ ] WHAT ARE GLOBAL AND REGION SPECIFIC SERVICES IN AWS ?

AWS STORAGE

- [ ] WHAT IS S3 AND HOW IS DATA STORED?
- [ ] THE SHARED RESPONSIBILITY MODEL AND SECURITY
- [ ] STORAGE TIERS AND PRICING
- [ ] GETTING DATA INTO AND OUT OF S3
- [ ] CREATE AND SECURE YOUR AWS ACCOUNT
- [ ] UPLOAD FILES TO BUCKETS USING THE AWS CONSOLE
- [ ] MOVE, COPY, DOWNLOAD, AND DELETE FILES
- [ ] CLASSIFYING YOUR BUCKETS AND OBJECTS WITH TAGS
- [ ] LIFECYCLE MANAGEMENT
- [ ] RETRIEVING OBJECTS FROM GLACIER
- [ ] MAKING BUCKETS OR OBJECTS PUBLIC WITH ACLS
- [ ] USING A BUCKET POLICY TO GRANT PUBLIC ACCESS
- [ ] USING A BUCKET POLICY TO GRANT ACCESS TO OBJECTS IN A BUCKET
- [ ] USING A BUCKET POLICY TO RESTRICT ACCESS BASED ON AN OBJECT



AWS S3 BASİCS

- [ ] USING A BUCKET POLICY TO RESTRICT ACCESS BASED ON AN OBJECT TAG
- [ ] HOW TO ENABLE VERSIONING AND ENCRYPTION
- [ ] HOW TO SET UP CROSS REGION REPLICATION FOR FURTHER REDUNDANCY

AWS IAM

1. INTRODUCTION TO AWS IAM

- [ ] WHAT IS AWS IAM?
- [ ] CORE CONCEPTS (USERS, GROUPS, ROLES, POLICIES)
- [ ] BENEFITS OF USING IAM FOR ACCESS MANAGEMENT

2. IAM USERS

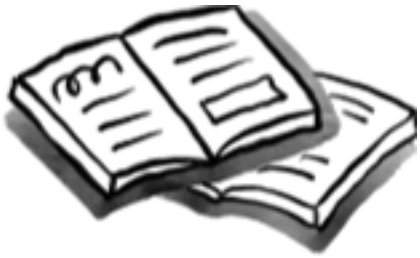
- [ ] CREATING AND MANAGING IAM
- [ ] USERS USER CREDENTIALS (PASSWORDS, ACCESS KEYS)
- [ ] BEST PRACTICES FOR MANAGING USER PERMISSIONSESS MANAGEMENT

3.IAM GROUPS

- [ ] CREATING AND MANAGING IAM
- [ ] GROUPS ADDING USERS TO GROUPS
- [ ] GROUP-LEVEL PERMISSIONS AND POLICIES

4. IAM ROLES

- [ ] WHAT ARE IAM ROLES?
- [ ] CREATING AND MANAGING ROLES
- [ ] ROLE-BASED ACCESS CONTROL (RBAC)
- [ ] USING ROLES WITH AWS SERVICES (E.G., EC2, LAMBDA)



AWS S3 BASİCS

5. IAM POLICIES

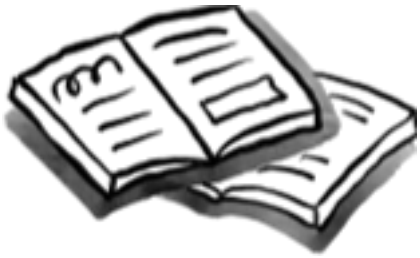
- [ ] WHAT ARE IAM POLICIES?
- [ ] AWS-MANAGED POLICIES VS. CUSTOMER-MANAGED POLICIES
- [ ] POLICY STRUCTURE (JSON)
- [ ] WRITING AND ATTACHING POLICIES
- [ ] POLICY EVALUATION LOGIC

6. AWS-MANAGED POLICIES

- [ ] OVERVIEW OF AWS-MANAGED POLICIES COMMON
- [ ] AWS-MANAGED POLICIES (E.G., ADMINISTRATORACCESS, READONLYACCESS)
- [ ] BEST PRACTICES FOR USING AWS-MANAGED POLICIES
- [ ] CUSTOMIZING AWS-MANAGED POLICIES

7. CUSTOM POLICIES

- [ ] CREATING CUSTOM IAM POLICIES
- [ ] POLICY ELEMENTS (ACTIONS, RESOURCES, CONDITIONS)
- [ ] USING POLICY VARIABLES
- [ ] TESTING AND DEBUGGING CUSTOM POLICIES
- [ ] BEST PRACTICES FOR CUSTOM POLICIES



AWS LAMBDA

1. INTRODUCTION TO AWS LAMBDA

- [ ] WHAT IS AWS LAMBDA?
- [ ] USE CASES FOR SERVERLESS COMPUTING
- [ ] BENEFITS OF USING AWS LAMBDA

2. LAMBDA FUNCTIONS

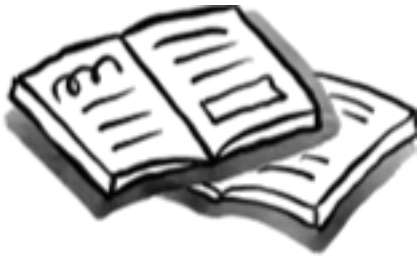
- [ ] CREATING LAMBDA FUNCTIONS
- [ ] LAMBDA FUNCTION LIFECYCLE
- [ ] WRITING YOUR FIRST LAMBDA FUNCTION
- [ ] LAMBDA FUNCTION CONFIGURATION (MEMORY, TIMEOUT, ENVIRONMENT VARIABLES)

3. EVENT SOURCES

- [ ] UNDERSTANDING EVENT-DRIVEN ARCHITECTURE INTEGRATING
- [ ] LAMBDA WITH VARIOUS AWS SERVICES (E.G., S3, DYNAMODB, SNS, SQS, API GATEWAY)
- [ ] CUSTOM EVENT SOURCES

4. AWS LAMBDA TRIGGERS

- [ ] CONFIGURING TRIGGERS FOR LAMBDA FUNCTIONS INVOKING
- [ ] LAMBDA FUNCTIONS SYNCHRONOUSLY AND ASYNCHRONOUSLY
- [ ] HANDLING RETRIES AND ERROR HANDLING



5. PERMISSIONS AND SECURITY

- [ ] AWS IDENTITY AND ACCESS MANAGEMENT (IAM) ROLES FOR LAMBDA
- [ ] LEAST PRIVILEGE PRINCIPLE
- [ ] RESOURCE-BASED POLICIES
- [ ] VPC INTEGRATION AND SECURITY GROUPS

6. LAMBDA LAYERS

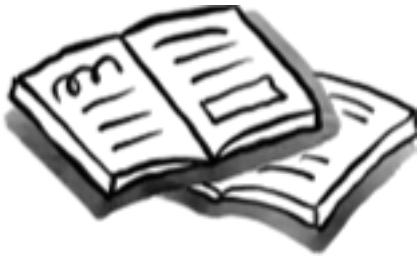
- [ ] WHAT ARE LAMBDA LAYERS?
- [ ] CREATING AND USING LAMBDA LAYERS
- [ ] BEST PRACTICES FOR USING LAYERS

7. ENVIRONMENT VARIABLES AND CONFIGURATION

- [ ] USING ENVIRONMENT VARIABLES IN LAMBDA FUNCTIONS
- [ ] SECRETS AND SENSITIVE DATA MANAGEMENT WITH AWS SECRETS MANAGER AND AWS SYSTEMS MANAGER PARAMETER STORE

8. LOGGING AND MONITORING

- [ ] LOGGING WITH AWS CLOUDWATCH LOGS
- [ ] METRICS AND MONITORING WITH AWS CLOUDWATCH
- [ ] SETTING UP CLOUDWATCH ALARMS
- [ ] TROUBLESHOOTING AND DEBUGGING



9. ERROR HANDLING AND RETRIES

- [ ] BUILT-IN ERROR HANDLING MECHANISMS
- [ ] RETRY STRATEGIES AND DEAD-LETTER QUEUES (DLQ)
- [ ] CUSTOM ERROR HANDLING

10. DEPLOYMENT AND VERSIONING

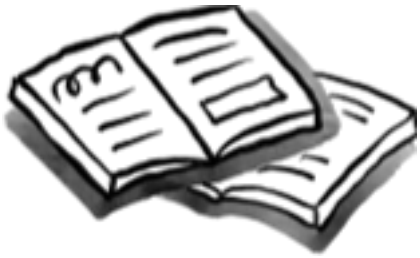
- [ ] DEPLOYING LAMBDA FUNCTIONS USING THE AWS MANAGEMENT CONSOLE, CLI, SDKS, AND CI/CD PIPELINES
- [ ] VERSIONING AND ALIASES B
- [ ] LUE/GREEN DEPLOYMENTS WITH ALIASES

11. PERFORMANCE OPTIMIZATION

- [ ] BEST PRACTICES FOR OPTIMIZING LAMBDA FUNCTION PERFORMANCE
- [ ] COLD STARTS AND WARM STARTS
- [ ] PROVISIONED CONCURRENCY

12. COST MANAGEMENT

- [ ] UNDERSTANDING LAMBDA PRICING
- [ ] COST OPTIMIZATION STRATEGIES
- [ ] MONITORING AND CONTROLLING LAMBDA USAGE COSTS

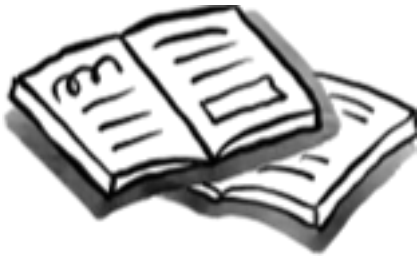


AWS EMR

- [ ] EMR BASICS
- [ ] CLUSTER MANAGEMENT
- [ ] DATA INGESTION
- [ ] DATA TRANSFORMATION
- [ ] DATA LOADING
- [ ] CLUSTER SECURITY
- [ ] AUTO SCALING
- [ ] CLUSTER OPTIMIZATION
- [ ] SPOT INSTANCES
- [ ] EMR BEST PRACTICES
- [ ] DATA WORKFLOW AUTOMATION
- [ ] INTEGRATION WITH OTHER AWS SERVICES

AWS REDHSİFT

- [ ] DATA WAREHOUSING CONCEPTS
- [ ] AMAZON REDSHIFT ARCHITECTURE ‘
- [ ] CLUSTER MANAGEMENT
- [ ] DATA LOADING
- [ ] COPY COMMAND
- [ ] INSERT STATEMENT
- [ ] AWS DMS
- [ ] DATA DISTRIBUTION: KEY, EVEN, AND ALL
- [ ] QUERY OPTIMIZATION
- [ ] DATA ENCRYPTION DATA COMPRESSION



AWS DYNAMODB

1. INTRODUCTION TO AMAZON DYNAMODB

- [ ] WHAT IS DYNAMODB?
- [ ] USE CASES FOR DYNAMODB
- [ ] KEY FEATURES AND BENEFITS

2. CORE CONCEPTS

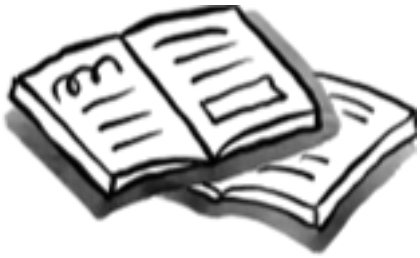
- [ ] TABLES, ITEMS, AND ATTRIBUTES
- [ ] PRIMARY KEYS (PARTITION KEYS AND SORT KEYS)
- [ ] SECONDARY INDEXES (GLOBAL SECONDARY INDEXES (GSI) AND LOCAL SECOND
- [ ] INDEXES (LSI))
- [ ] DYNAMODB STREAMS

3. SETTING UP DYNAMODB

- [ ] CREATING AND MANAGING TABLES
- [ ] UNDERSTANDING CAPACITY MODES (PROVISIONED VS. ON-DEMAND)
- [ ] DEFINING PRIMARY KEYS AND INDEXES
- [ ] SETTING UP DYNAMODB STREAMS

4. DATA MODELING

- [ ] DESIGNING EFFICIENT TABLE SCHEMAS
- [ ] UNDERSTANDING SINGLE-TABLE DESIGN
- [ ] USING PRIMARY KEYS AND SECONDARY INDEXES EFFECTIVELY
- [ ] BEST PRACTICES FOR DATA MODELING IN DYNAMODB

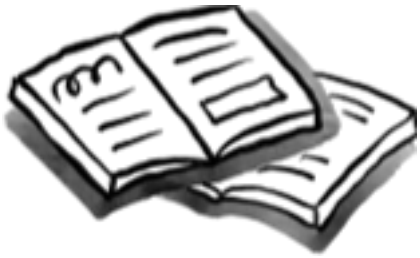


5. READING AND WRITING DATA

- [ ] USING THE DYNAMODB API (PUTITEM, GETITEM, UPDATEITEM, DELETEITEM)
- [ ] BATCH OPERATIONS (BATCHGETITEM, BATCHWRITEITEM)
- [ ] QUERYING AND SCANNING TABLES
- [ ] UNDERSTANDING CONDITIONAL OPERATIONS

6. INDEXES

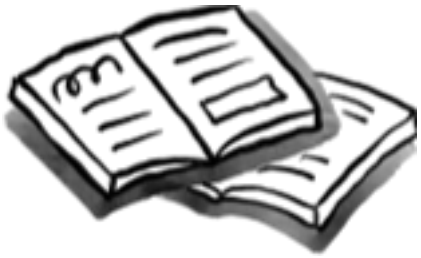
- [ ] CREATING AND MANAGING GLOBAL SECONDARY INDEXES (GSI)
- [ ] CREATING AND MANAGING LOCAL SECONDARY INDEXES (LSI)
- [ ] USE CASES AND BEST PRACTICES FOR USING INDEXES QUERYING DATA WITH INDEXES



AWS GLUE

- [ ] WHAT IS GLUE ?
- [ ] DATA CATALOG
- [ ] CRAWLERS
- [ ] DATA LAKE AND DATA WAREHOUSE INTEGRATION
- [ ] DATA BREW TRANSFORMATIONS JOB AUTHORING & DEVELOPMENT
- [ ] DATA SOURCE CONNECTORS
- [ ] TARGET CONNECTORS
- [ ] SERVERLESS EXECUTION
- [ ] MONITORING AND LOGGING
- [ ] SECURITY AND DATA ENCRYPTION
- [ ] ERROR HANDLING AND RETRY MECHANISMS
- [ ] DATA QUALITY AND VALIDATION
- [ ] DATA VERSIONING

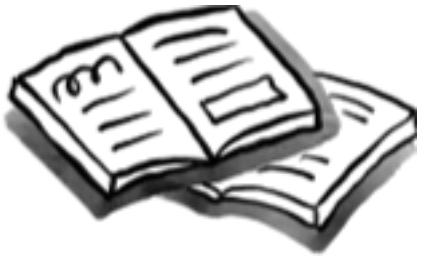
Module - 10



DATA STRUCTURES AND ALGORİTHMS

- [ ] DATATYPES
- [ ] OPERATORS
- [ ] CONDITIONAL STATEMENTS
- [ ] LOOPS
- [ ] ARRAYS
- [ ] STRINGS
- [ ] SEARCHING
- [ ] LINEAR SEARCH
- [ ] BINARY SEARCH
- [ ] SORTING
- [ ] BUBBLE SORT
- [ ] MERGE SORT
- [ ] LINKED LIST
- [ ] SINGLE LINKED LIST
- [ ] DOUBLE LINKED LIST
- [ ] STACK
- [ ] QUEUE
- [ ] TIME COMPLEXITY
- [ ] SPACE COMPLEXITY

Module - 11



AIRFLOW

SECTION 1: DOCKER ESSENTIALS FOR AIRFLOW SETUP

- [ ] DOCKER INSTALLATION (DOCKER DESKTOP)
- [ ] DOCKER INSTALLATION (DOCKER TOOLBOX)
- [ ] WRITING PROJECT COMPOSE FILE
- [ ] UNDERSTANDING COMPOSE FILES
- [ ] UNDERSTANDING OTHER DIRECTORIES

SECTION 2: AIRFLOW INSTALLATION & FIRST LOOK

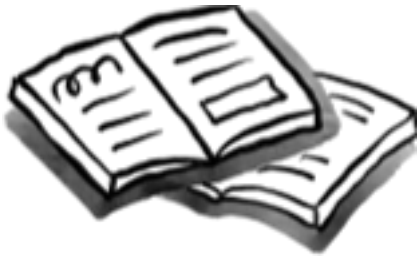
- [ ] TOPIC DURATION
- [ ] AIRFLOW INSTALLATION
- [ ] FIRST LOOK OF AIRFLOW UI
- [ ] RUNNING DEFAULT DAG IN UI

WHY AIRFLOW & ARCHITECTURE DEEP DIVE

- [ ] WHY MOVE TO AIRFLOW
- [ ] ARCHITECTURE OF AIRFLOW
- [ ] LIFE CYCLE OF A TASK

DAG CREATION & EXECUTION

- [ ] UNDERSTANDING DAG DEFINITION FILE
- [ ] DAG FILE EXECUTION
- [ ] CREATE A DAG (1 QUESTION)
- [ ] WRITING DAG CONTINUED
- [ ] RUNNING PROJECT DAG IN UI
- [ ] RUNNING PROJECT DAG IN AIRFLOW CLI



AIRFLOW

AIRFLOW CLI & PYTHON CONTEXT MANAGER

- [ ] AIRFLOW CLI COMMANDS - PART 1
- [ ] AIRFLOW CLI COMMANDS - PART 2
- [ ] 'WITH' - CONTEXT MANAGER

AIRFLOW CORE CONCEPTS – OPERATORS & EXECUTORS

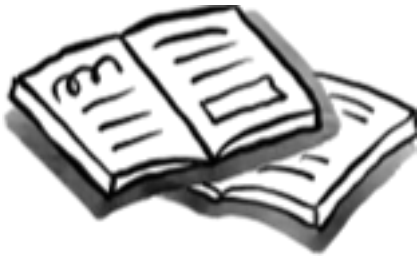
- [ ] WHAT ARE OPERATORS
- [ ] DUMMY OPERATOR
- [ ] WHAT ARE EXECUTORS
- [ ] SEQUENTIAL EXECUTOR
- [ ] LOCAL EXECUTOR

AIRFLOW UI – ADVANCED USAGE

- [ ] WHY MOVE TO AIRFLOW
- [ ] ARCHITECTURE OF AIRFLOW
- [ ] LIFE CYCLE OF A TASK

AIRFLOW UI – ADVANCED USAGE

- [ ] CREATING CONNECTIONS IN UI
- [ ] VARIABLES IN AIRFLOW

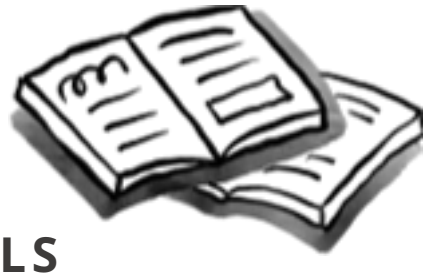


AIRFLOW

INTEGRATING AIRFLOW WITH SPARK AND HIVE

- ☐ TOPIC DESCRIPTION
- ☐ OVERVIEW OF SPARK & HIVE INTEGRATION
- ☐ WHY INTEGRATE SPARK AND HIVE WITH AIRFLOW
- ☐ SETTING UP CONNECTIONS (HIVE & SPARK)
- ☐ USE AIRFLOW UI TO CREATE SPARK_DEFAULT AND HIVE_DEFAULT CONNECTIONS
- ☐ USING SPARKSUBMITOPERATOR TRIGGER PYSPARK/SCALA JOBS FROM AIRFLOW
- ☐ USING HIVEOPERATOR EXECUTE HIVE QUERIES USING HQL
- ☐ TRIGGER HIVE TABLES FROM SPARK JOBS ORCHESTRATE HIVE JOBS POST SPARK PROCESSING
- ☐ CHAINING SPARK & HIVE TASKS IN DAG BUILD REAL-TIME DAGS WITH TASK DEPENDENCIES BETWEEN SPARK AND HIVE
- ☐ HANDLING ERRORS & DEPENDENCIES RETRY LOGIC, FAILURE ALERTS, TRIGGER RULES FOR SPARK/HIVE JOBS
- ☐ MONITORING SPARK & HIVE JOBS USE AIRFLOW LOGS, SPARK UI, AND HIVE CLI LOGS
- ☐ USE CASE: END-TO-END PIPELINE BUILD A PROJECT: INGEST CSV → SPARK TRANSFORMATION → HIVE LOAD → NOTIFICATION

Module - 12



1. DATABRICKS FUNDAMENTALS

INTERNAL TOPICS:

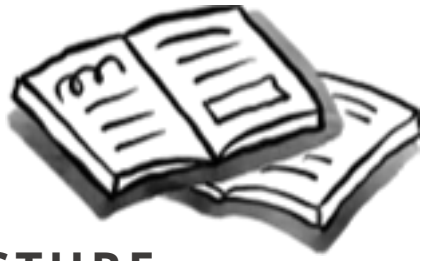
- [ ] INTRODUCTION TO DATABRICKS AND ITS ECOSYSTEM
- [ ] HISTORY OF DATABRICKS AND APACHE SPARK
- [ ] OVERVIEW OF THE UNIFIED ANALYTICS PLATFORM
- [ ] NAVIGATING THE DATABRICKS UI
- [ ] WORKSPACE STRUCTURE: REPOS, NOTEBOOKS, WORKFLOWS, JOBS
- [ ] DATABRICKS CLI AND REST API INTRODUCTION

CLUSTERS

- [ ] CLUSTER TYPES: ALL-PURPOSE, JOB, POOL-BACKED CLUSTERS
- [ ] CLUSTER CONFIGURATION (AUTOSCALING, SPOT VS ON-DEMAND)
- [ ] NOTEBOOKS AND CELLS
- [ ] MAGIC COMMANDS (%SQL, %PYTHON, %SCALA)
- [ ] RUNNING AND SCHEDULING NOTEBOOKS

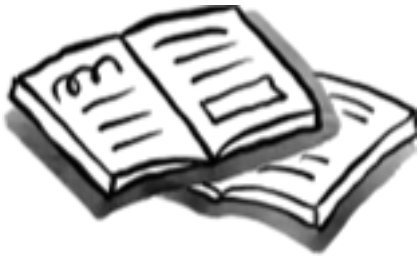
DATABRICKS RUNTIME (DBR)

- [ ] VERSIONS AND THEIR USE CASES (DBR ML, DBR GENOMICS, DBR PHOTON)
- [ ] WORKSPACE AND USER MANAGEMENT
- [ ] ROLES AND PERMISSIONS
- [ ] ADMIN CONSOLE AND IDENTITY MANAGEMENT (SCIM)



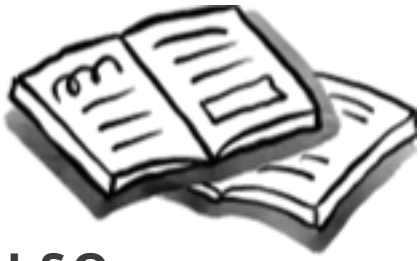
2. DATA LAKEHOUSE ARCHITECTURE

- [ ] TRADITIONAL DATA WAREHOUSE VS DATA LAKE VS
- [ ] LAKEHOUSE
- [ ] COMPONENTS OF THE LAKEHOUSE ARCHITECTURE:
- [ ] STORAGE LAYER (E.G., ADLS, S3)
- [ ] METADATA LAYER (UNITY CATALOG, HIVE METASTORE)
- [ ] COMPUTE LAYER (SPARK CLUSTERS, PHOTON)
- [ ] DELTA LAKE AS THE FOUNDATION OF LAKEHOUSE
- [ ] MEDALLION ARCHITECTURE IN LAKEHOUSE
- [ ] BRONZE: RAW INGESTED DATA
- [ ] SILVER: CLEANED AND FILTERED
- [ ] GOLD: AGGREGATED BUSINESS-LEVEL DATA
- [ ] ACID TRANSACTIONS IN DELTA LAKE
- [ ] SCHEMA EVOLUTION AND ENFORCEMENT
- [ ] TIME TRAVEL & DATA VERSIONING
- [ ] DATA LINEAGE AND GOVERNANCE IN LAKEHOUSE



3. DATABRICKS UNITY CATALOG

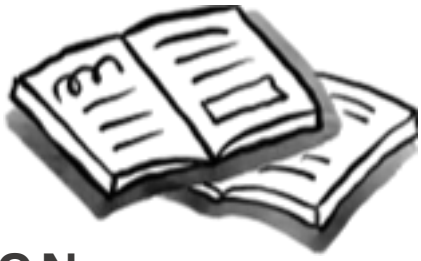
- [ ] WHAT IS UNITY CATALOG AND ITS BENEFITS
- [ ] HIERARCHICAL MODEL:
- [ ] METASTORE > CATALOG > SCHEMA > TABLE/VIEW/FUNCTION
- [ ] SECURING DATA WITH UNITY CATALOG
- [ ] TABLE ACLS
- [ ] ROW- AND COLUMN-LEVEL SECURITY
- [ ] MASKING POLICIES
- [ ] MANAGING UNITY CATALOG
- [ ] ASSIGNING METASTORES TO WORKSPACES
- [ ] GRANTING ACCESS WITH GRANT/REVOKE
- [ ] MANAGING SERVICE PRINCIPALS AND IDENTITIES
- [ ] UNITY CATALOG VS HIVE METASTORE
- [ ] DATA LINEAGE TRACKING
- [ ] INTEGRATION WITH EXTERNAL TOOLS (POWER BI, TABLEAU, LOOKER)



4. DATABRICKS AUTOLOADER (ALSO CALLED AUTO CATALOG OR INCREMENTAL LOADER)

INTERNAL TOPICS:

- [ ] WHAT IS AUTO LOADER AND WHY IT'S USED
- [ ] SUPPORTED FORMATS (JSON, PARQUET, CSV, AVRO, ETC.)
- [ ] FILE NOTIFICATION MODE VS DIRECTORY LISTING MODE
- [ ] CLOUDFILES CONFIGURATION AND SCHEMA EVOLUTION
- [ ] INCREMENTAL DATA PROCESSING USING AUTO LOADER
- [ ] COMBINE WITH STRUCTURED STREAMING
- [ ] FAULT TOLERANCE AND CHECKPOINTING
- [ ] LOADING INTO BRONZE LAYER (MEDALLION MODEL)
- [ ] USING AUTOLOADER WITH UNITY CATALOG

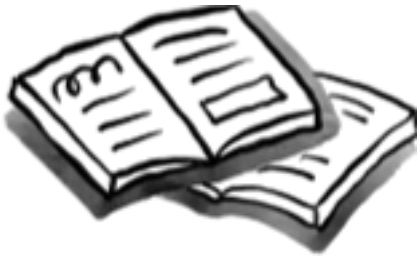


5. PERFORMANCE OPTIMIZATION TECHNIQUES IN DATABRICKS

INTERNAL TOPICS:

- ☐ CLUSTER TUNING AND SELECTION
- ☐ AUTOSCALING, WORKER TYPES, INSTANCE FAMILIES
- ☐ DATA SKEW HANDLING TECHNIQUES
- ☐ SALTING, BROADCASTING, REPARTITIONING
- ☐ CACHING AND PERSISTENCE (CACHE(), PERSIST())
- ☐ COST-BASED OPTIMIZATION (CBO)
- ☐ ADAPTIVE QUERY EXECUTION (AQE)
- ☐ Z-ORDERING AND FILE COMPACTION
- ☐ PARTITIONING STRATEGIES FOR PERFORMANCE
- ☐ JOB PERFORMANCE ANALYSIS WITH SPARK UI AND GANGLIA
- ☐ PHOTON ENGINE OPTIMIZATION
- ☐ WHEN AND HOW TO USE IT
- ☐ SHUFFLE MANAGEMENT & BROADCAST JOINS
- ☐ WRITING EFFICIENT SQL IN DATABRICKS

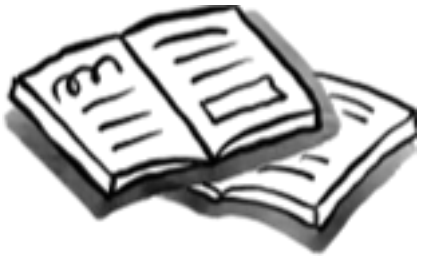
6. DELTA LIVE TABLES (DLT) + TYPES OF CLUSTERS



INTERNAL TOPICS:

- [ ] INTRODUCTION TO DELTA LIVE TABLES
- [ ] DIFFERENCE BETWEEN DLT AND NOTEBOOKS
- [ ] SQL VS PYTHON DLT PIPELINES
- [ ] PIPELINE MODES
- [ ] TRIGGERED, CONTINUOUS, DEVELOPMENT
- [ ] TABLE TYPES IN DLT
- [ ] STREAMING TABLE
- [ ] MATERIALIZED VIEW
- [ ] LIVE TABLE
- [ ] QUALITY CHECKS AND EXPECTATIONS (DATA QUALITY RULES)
- [ ] EXPECT_OR_DROP, EXPECT_OR_FAIL
- [ ] SCHEMA ENFORCEMENT
- [ ] MONITORING DLT PIPELINES
- [ ] ORCHESTRATION USING WORKFLOWS
- [ ] DEPLOYMENT OF DLT (API, UI, CLI)
- [ ] CLUSTER TYPES DEEP DIVE:
- [ ] INTERACTIVE (ALL-PURPOSE) VS JOB CLUSTERS
- [ ] PHOTON-ENABLED VS NON-PHOTON
- [ ] SPOT VS ON-DEMAND VS RESERVED
- [ ] POOL CLUSTERS VS NON-POOLED
- [ ] CLUSTER AUTOSCALING VS FIXED

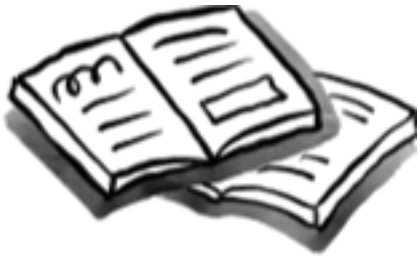
Module - 13



PROJECTS

- [ ] HEALTH-CARE SECTOR USE CASE
- [ ] RETAIL USE CASE PROJECT
- [ ] FINANCIAL USE CASE PROJECT

2.Placement Assistance Program



- ☐ 1) LINKEDIN PROGRAM TO GRAB OPPORTUNTIES
- ☐ 2)EVERY SATURDAY 2PM TO 5PM WE HAVE A PROFILE REVIEW SESSION
- ☐ 3)EVERY SUNDAY WE HAVE CONTENT STRATEGY CLASS
- ☐ 4)NAUKRI OPTIMIZATION SESSION
- ☐ 5)RESUME BUILDING SESSION

3. INTERVIEW PREPARATION PHASE:

- ☐ 1)REVISION
- ☐ 2)EXAMS 40+ SUBJECTIVE EXAMS
- ☐ 3)MOCK-INTERVIEWS
- ☐ 4)30 HOURS OF INTERVIEW PREPARATION
- ☐ 5)TOPICS COVERED
  - 1)HDFS
  - 2)HIVE
  - 3)SQL
  - 4)PYTHON AND SCALA
  - 5)SPARK
  - 6)AWS
  - 7)PROJECT RELATED ACTIVITIES